bblt Version 3.1 User's Manual

T. Tedford, E. Gonzalez, and C. Hannah

Ocean Sciences Division
Maritimes Region
Fisheries and Oceans Canada

Bedford Institute of Oceanography
P.O. Box 1006
Dartmouth, Nova Scotia
Canada B2Y 4A2

2001

Canadian Technical Report of Hydrography and Ocean Sciences 213

Fisheries and Oceans Canada | Pêches et Océans Canada

Canada

101-05711/

Canadian Technical Report of
Hydrography and Ocean Sciences 213

2001

bblt Version 3.1 User's Manual

by

Ted Tedford, Elizabeth Gonzalez, Charles Hannah

Ocean Sciences Division
Maritimes Region
Fisheries and Oceans Canada

Bedford Institute of Oceanography
P.O. Box 1006
Dartmouth, N.S.
Canada B2Y 4A2

©Minister of Supply and Services Canada 2001
Cat. No. Fs 97-18/213E ISSN 0711-6764

Correct citation for this publication:

Tedford, T., Gonzalez, E. and Hannah, C. G. 2001. `bblt` Version 3.1 User's Manual. Can. Tech. Rep. Hydrogr. Ocean Sci. 213: v + 48 pp.

Table of Contents

List of Figures iv

List of Tables iv

Abstract/Résumé v

1 Introduction 1

2 Discussion of `bblt` Concepts 3
- 2.1 Transport and Dispersion . . . 3
- 2.2 Overview of `bblt` . . . 3
- 2.3 Vertical Mixing . . . 7
- 2.4 Vertical Shear . . . 8

3 Input Parameters 8

4 Model Output and Derived Quantities 13

5 Software Overview 14
- 5.1 `bblt v3.1` . . . 15
- 5.2 Extensions of Local `bblt` since Version 1.0 . . . 18

6 Sample `bblt` run 19
- 6.1 Pseudocode Input . . . 19
- 6.2 Input . . . 24
- 6.3 Output . . . 28
- 6.4 Plots . . . 32

7 Final Comments 32

References 34

A Logarithmic Sublayer 36

B List of Files 39

C Main Module 40

List of Figures

1 Schematic flow chart for the `bblt` models 2
2 Shear dispersion 4
3 Shear dispersion of a vertical line of particles 5
4 Schematic of shearing and shuffling packets 7
5 Schematic time line 11
6 The vertical coordinate scheme 12
7 Graphical representation of output from example case 33

List of Tables

1 List of input parameters for `bblt` v3.1. 9
2 List of symbols 10
3 Outline of the algorithm for `bblt v3.1`. 16
4 Pseudocode representation of the bblt input file 20
5 Sample `bblt` input file 25
6 Sample 'log_layer' file 26
7 Beginning of an observed currents file 27
8 Sample 'current_heights' file 28
9 First 10 lines of `sedparams.out` 29
10 First 10 lines of `stats.out` 30
11 First 10 lines of `pack.out` 31
12 First 10 lines of `cnstr.out` 31

Abstract

Tedford, T., Gonzalez, E. and Hannah, C. G. 2001. `bblt` Version 3.1 User's Manual. Can. Tech. Rep. Hydrogr. Ocean Sci. 213: v + 48 pp.

The benthic boundary layer transport model, `bblt`, was developed to predict the transport and dispersion of suspended particulate drill waste in the benthic boundary layer. This user's guide is a description of `bblt`, version 3.1, which assumes a spatially homogeneous environment and allows for bulk releases of sediment and some restricted continuous release scenarios. There is a discussion of the basic concepts, a description of the algorithm, a description of the required input parameters and forcing functions, a brief discussion of the basic model output and an overview of the software. There is also a detailed description of a sample run of `bblt v3.1`.

Résumé

Tedford, T., Gonzalez, E. and Hannah, C. G. 2001. `bblt` Version 3.1 User's Manual. Can. Tech. Rep. Hydrogr. Ocean Sci. 213: v + 48 pp.

Un modèle de transport dans la couche limite benthique, `bblt`, a été développé pour prévoir le transport et la dispersion des rejets de forage particulaire en suspension. Cet manuel d'utilisation est une description du `bblt`, version 3.1, qui assume un environnement homogène dans l'espace et permet des décharges de sédiment en bloc et certain scénarios de décharge continue. Il y a une discussion des concepts base, une description de l'algorithme, une description des paramètres nécessaires en entrée et des fonctions de forçage, une brève discussion de la sortie du modèle et une description sommaire du logiciel. Il y a aussi une description détaillée d'une simulation de base du `bblt v3.1`.

1 Introduction

The benthic boundary layer transport model, `bblt`, was developed to predict the transport and dispersion of suspended particulate drill waste in the benthic boundary layer. The motivating application was the prediction of critical impact zones around exploration and production drilling platforms in regions where there is potential for negative impacts on commercially important benthic organisms, e.g. sea scallops on Georges Bank (Gordon et al., 1993, 2000; Muschenheim et al., 1995). There is particular interest in the dispersion of flocculated drill mud fines, which can have settling velocities which allow them to be concentrated near the seafloor (Muschenheim et al., 1995; Muschenheim and Milligan, 1996). In this report, 'drill waste' and 'sediment' are both used to refer to 'flocculated drill mud fines.'

The basic idea of the `bblt` family of models is illustrated in Figure 1. Estimates of the current profile ($\underline{\mathbf{u}}$) and bottom stress ($\underline{\tau}_b$) are combined with estimates of the vertical profiles of sediment concentration (generally a function of settling velocity and bottom stress) and vertical mixing to generate estimates of drift and dispersion. These ideas are implemented as follows. The sediment load is partitioned into discrete pseudo-particles or 'packets' each with mass m and settling velocity w_s. These packets are advected horizontally and mixed vertically. The overall (horizontally-averaged) vertical distribution of the sediment is governed by a specified quasi-equilibrium concentration profile which is used to derive a probability density function for the vertical position of the packets. Vertical mixing is represented by random exchange (shuffling) of the packets, which is controlled by a specified vertical mixing time scale. The key modelling parameters are the settling velocity w_s, the friction velocity u_*, and the shuffling time scale t_m. For a given velocity profile, these parameters determine the results.

This user's guide is a description of `bblt`, version 3.1, which assumes a spatially homogeneous environment ('local' version) and allows for bulk releases of sediment and some restricted continuous release scenarios. Horizontal variations in water depth, currents and bottom stress are ignored (Hannah et al., 1995, 1996). This simplification results in a major reduction in computational demand, while retaining the effects of vertical and temporal variations in the current which are generally the primary factors in short-term dispersion and transport at local positions.

This guide contains a discussion of the basic concepts and a description of the algorithm (Section 2), a description of the required input parameters and forcing functions (Section 3), a brief discussion of the basic model output (Section 4) and an overview of the software (Section 5). There is also a detailed description of a sample run of `bblt v3.1` (Section 6).

The technical development of `bblt` has been documented in Hannah et al. (1995, 1996, 1998), Loder et al. (2001) and Xu et al. (2000). The application of `bblt` as one component of an impacts zone assessment model for Georges Bank is documented in Loder et al.

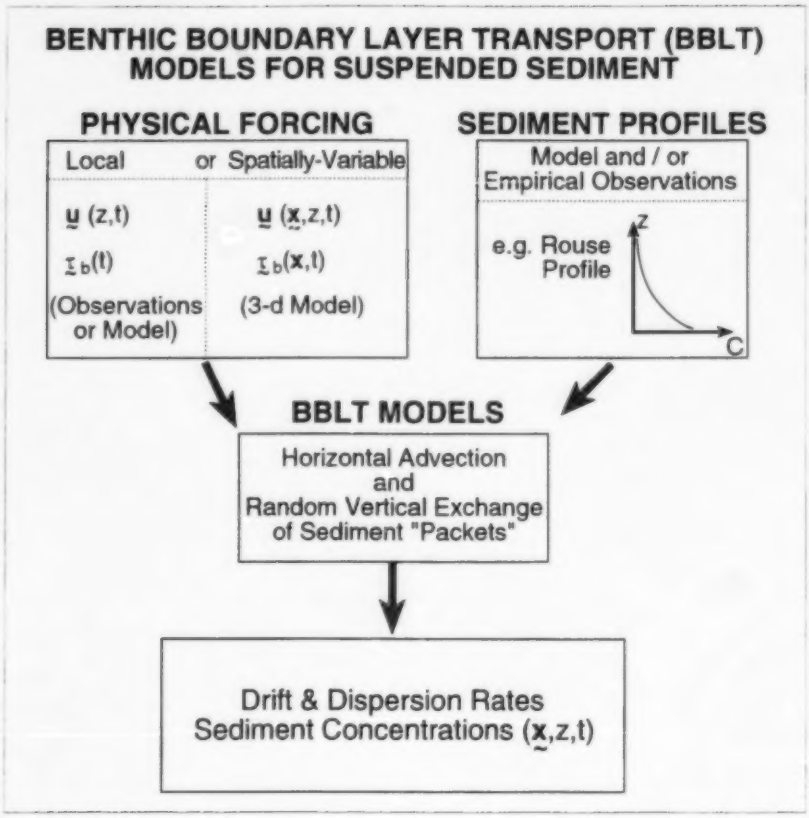

Figure 1: Schematic flow chart for the `bblt` models. Quantitative information on currents, bottom stress and the vertical structure of suspended sediment is used in the models to estimate sediment drift and dispersion rates.

(2001), Gordon et al. (2000), and Boudreau et al. (1999). Loder et al. (2001) is a useful companion to this user's guide as it contains both a technical description of `bblt` and an extensive application of `bblt` to Georges Bank. Hannah et al. (1995) is a useful description of the basic processes represented in `bblt`.

The current profile in `bblt`, version 3.1, can come from current meter observations or numerical models. The model simply requires time series of velocity at user specified depths. This guide will use the phrase 'current meter' as shorthand for 'the time series of velocity.'

2 Discussion of bblt Concepts

2.1 Transport and Dispersion

The conceptual model of suspended sediment transport that forms the foundation of bblt is illustrated in Figure 2. Consider an initial distribution of sediment (time 1). The currents and bottom stress increase and suspend additional sediment (time 2). The distribution of suspended sediment is stretched in the horizontal by vertical shear in the currents and redistributed vertically by vertical mixing (time 3). Finally, the currents relax and sediment settles back to the bottom, with a broader distribution and new horizontal center of mass (time 4).

The primary dispersion mechanism active in the model is shear dispersion, in our case, horizontal dispersion due to the interaction of vertical mixing and vertical shear. Figure 3 illustrates the idea that in a purely periodic flow (here, rectilinear), an initial line of particles can be smeared out due to the interaction of vertical mixing and vertical shear (Fischer et al., 1979). In the absence of vertical mixing, the vertical line of particles is tilted back and forth over the tidal period but there is no net dispersion. On the other hand, in the presence of vertical mixing, particles are displaced vertically from the tilted line, resulting in horizontal dispersion during each periodic tilting.

2.2 Overview of bblt

In the formulation of bblt, there was an implicit assumption that the sediment particles have a significant settling velocity and therefore, for a bottom source of sediment, the concentration profile is a maximum near the bottom and decreases monotonically towards the surface. This assumption is made explicit by our use of a modified Rouse concentration profile, and is supported by available observations from Sable Island Bank (Muschenheim et al. 1995). The concentration profile $c(z)$ used in bblt is

$$c(z) = c_0(h_{ref}/z)^p \tag{1}$$

where c_0 is the concentration at the reference height $z = h_{ref}$, $z = 0$ at the bottom and $p = w_s/(\kappa u_*)$ where w_s is the characteristic settling velocity, $\kappa = 0.4$ is the von Karman constant and u_* is the friction velocity (related to the bottom stress by $u_* = \sqrt{\tau_b/\rho}$ where τ_b is the magnitude of the bottom stress and ρ is the density of sea water). The reference height is taken as the lower boundary of the suspended sediment profile with a typical value of $h_{ref} = 0.0035$ m (e.g. Loder et al., 2001). Particle motion below h_{ref} is assumed to be negligible and is ignored in bblt.

The linear bottom stress law used to calculate u_* in the original version of bblt has been

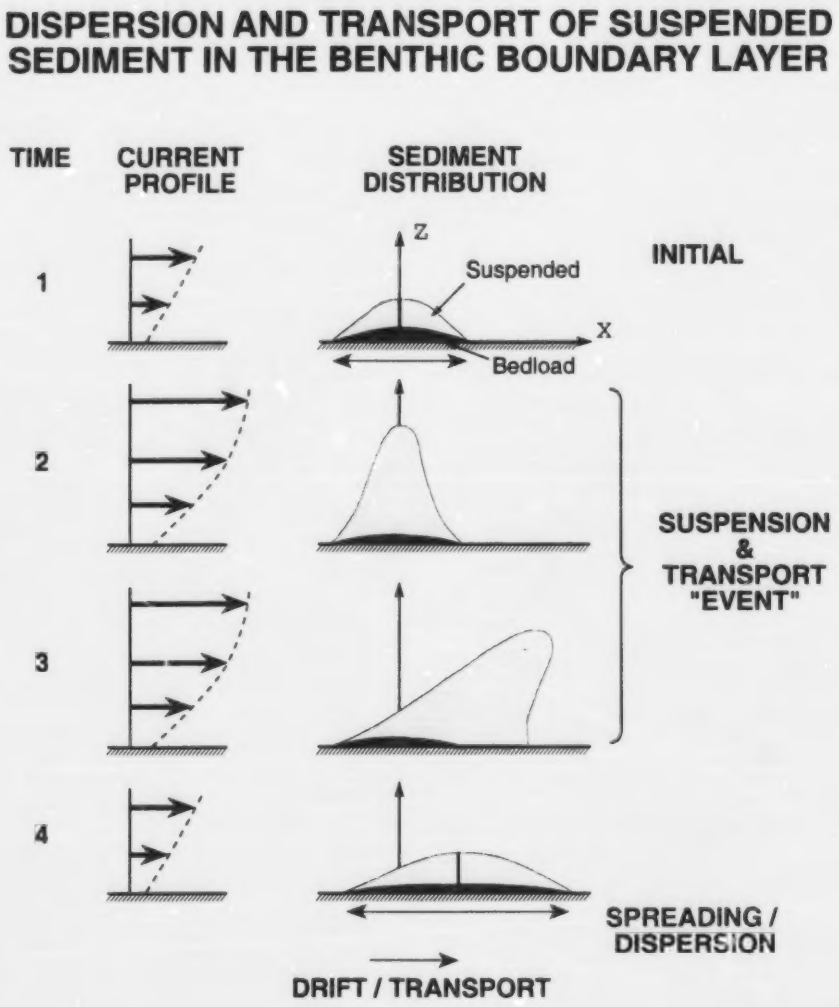

Figure 2: Schematic illustrating how sediment suspension and vertical shear in horizontal currents in the benthic boundary layer lead to horizontal drift and shear dispersion.

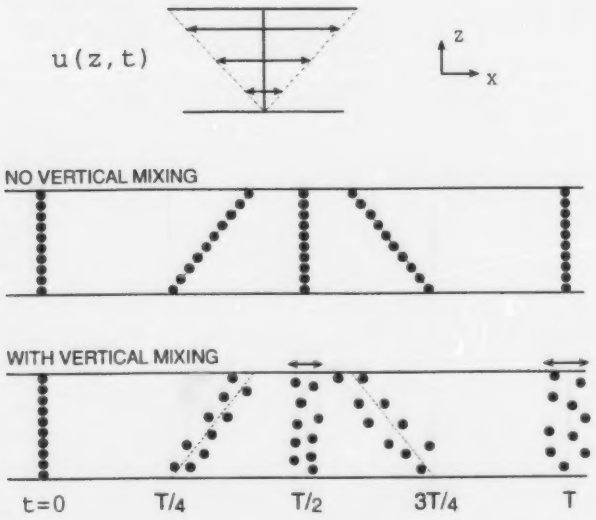

Figure 3: Schematic of shear dispersion of a vertical line of particles in a horizontal tidal current with vertical shear. Vertical mixing is required for horizontal dispersion over a tidal period. The case shown (a tilt in each direction during each tidal period) assumes peak currents at $t = 0$ ($u = \cos\omega t$).

replaced by a polynomial stress law:

$$u_*^2 = \sum_{i=0}^{I} a_i |\mathbf{u}_b|^i \tag{2}$$

where $|\mathbf{u}_b|$ is the magnitude of the observed near-bottom current and the a_i are user-specified bottom friction parameters. Most recent `bblt` applications (e.g. Hannah et al., 1998) have used a simple quadratic law,

$$u_*^2 = C_d |\mathbf{u}_b|^2, \tag{3}$$

where $C_d = a_2$ is a drag coefficient. The quadratic law generates larger maximum u_* and smaller minimum u_* than the linear law. This can have a substantial effect if there are large temporal variations in the near-bottom velocity and hence in the vertical extent of the sediment profile over a tidal cycle. The mean drift appears to be more sensitive to the details of the stress law than the dispersion, because of Stokes velocity-like effects from the sediment's varying vertical position in the presence of variable vertical shear.

A feature of `bblt` is the use of discrete pseudo-particles or "packets" rather than continuous concentrations to model the transport problem. The mass of material to be modelled, M, is divided into N packets each with a mass $m = M/N$ and a common user specified settling velocity w_s. The model is formulated in Lagrangian coordinates, with the packet

positions, $X_n(t)$, $Y_n(t)$ and $Z_n(t)$, as the basic output. There is no grid in the horizontal or the vertical. It is important to note the packets are mathematical constructs with different vertical movement properties than real sediment particulates or flocs; rather, the net effect of their movement is intended to represent real-ocean shear dispersion. Output quantities, such as concentration, are derived with a user defined grid.

The movement of the packets is divided into two parts: the vertical distribution and the horizontal transport. Given a time-dependent velocity field, an advection time step Δt_a, and a distribution of packets, the horizontal advection of the packets is straight-forward. The difficulties, both conceptually and numerically, are with the vertical distribution.

Vertical motions are not modelled explicitly; they are parameterized in terms of the semi-empirical Rouse profile (1) taken from the suspended sediment literature. We assume that the important features of the small-scale interactions between the sediment, the bottom, and the turbulence in the water column are captured by the vertical profile of the sediment concentration. This approach is consistent with the form of most available quantitative information on suspended sediment transport.

For our purposes, the concentration profile is a mathematical construct which contains the dependence of the vertical distribution of the sediment on the local bottom stress and the settling velocity of the sediment particles. We interpret the concentration profile as a probability distribution which provides a way to assign the vertical positions to the packets consistent with the concentration profile. We assume that the profile is an equilibrium one, with no explicit time dependence. Time dependence occurs only through the dependence of the concentration profile on time varying bottom stress.

The basic assumption is that the amount of material in suspension is known, the problem is reduced to distributing the packets in the vertical such that the concentration profile (1) is satisfied. The concentration profile is used to determine the vertical positions of the packets as follows. Define $F(z)$ as the fraction of the mass which exists between the reference height h_{ref} and the depth z:

$$F(z) = \int_{href}^{z} \hat{c}(z')\, \mathrm{d}z' \tag{4}$$

where $\hat{c}(z)$ is the normalized concentration profile,

$$\int_{href}^{hmax} \hat{c}(z)\, \mathrm{d}z \equiv 1, \tag{5}$$

and h_{max} is the maximum height of the sediment profile. Assign each packet n a rank α_n uniformly distributed in the interval $(0, 1)$. The vertical position of any packet is the solution z_n to the equation $F(z_n) = \alpha_n$. Thus, we convert a uniform distribution to a Rouse distribution; a packet with $\alpha = 0$ is positioned at $z = h_{ref}$ and a packet with $\alpha = 1$ is positioned at $z = h_{max}$. During integration $F(z)$ is updated at each time step

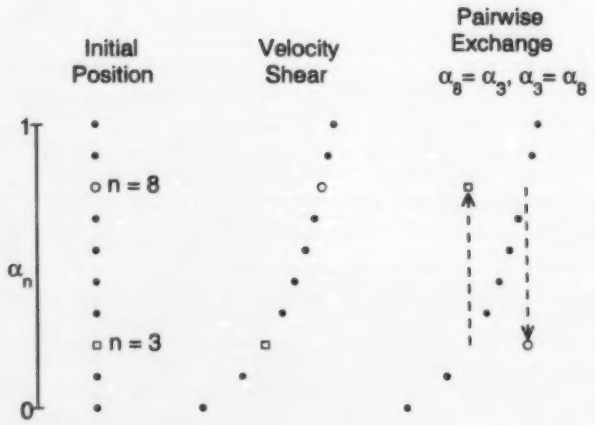

Figure 4: Schematic of shearing and shuffling of a vertical line of packets in a horizontal current. The packet ranks (α_n) are uniformly distributed in the interval (0,1). The exchange of ranks is performed at each advection time step, Δt_a, for $N\Delta t_a/t_m$ randomly selected pairs of packets. A packet with $\alpha = 1$ is located at h_{max} and a packet with $\alpha = 0$ is located at h_{ref}.

to reflect changes in the concentration profile $c(z)$. As the bottom stress changes the concentration profile expands and contracts and the vertical coordinates of the packets change in concert. This general procedure for computing vertical positions means that `bblt` can be modified to use other concentration profiles without changing the basic structure of the model.

2.3 Vertical Mixing

Vertical mixing is parameterized through the time-varying Rouse profile and random vertical shuffling of the packets. The packet shuffling consists of pair-wise exchanges of the α_n as shown in Figure 4. After each advection time step, Δt_a, the α_n are exchanged for $N\Delta t_a/t_m$ randomly selected pairs of packets, where N is the total number of packets and t_m is a specified mixing time scale. The shuffling is intended to extract the net effect of the shear dispersion processes without having to explicitly model the motions of individual sediment particles. Other shuffling algorithms can be implemented without changing the basic structure of `bblt`.

With a time-varying bottom stress, vertical redistribution of the packets also takes place through expansion and contraction of the profile. If there is no shuffling then the packets maintain their relative vertical positions but their absolute vertical position can change

as the bottom stress changes. On the other hand if the bottom stress is constant then an individual packet can only change its vertical position due to the shuffling of the α_n. On a global basis the equilibrium profile is maintained perfectly, but local deviations can occur.

The use of equilibrium concentration profiles requires that the time scale for the vertical mixing of the suspended sediment be shorter than the time scale for changes in the bottom stress. For the purposes of this work we assume that the velocity field (and the bottom stress) evolves on tidal time scales, i.e., order of hours.

2.4 Vertical Shear

The representation of the vertical shear in the currents is important because this velocity shear is a critical factor in shear dispersion which is the core process in `bblt`.

In the interior of the water column, it is assumed that the velocity shear between the current meters is adequately captured by linear interpolation. If this is not sufficient, then prior knowledge must be used to create synthetic current time series (between the current meters) and these used as additional input to `bblt`.

The details of the near bottom velocity profile are important because most of the tidal current shear occurs here. A logarithmic profile is used to calculate the velocity below the lowest (nearest to the bottom) current meter (this replaces the linear profile used in `bblt` v1). The logarithmic layer is an approximation of the turbulent boundary layer which generally extends a few metres above the seafloor. The lowest current observation should come from within this layer. Details of the implementation are provided in Appendix A.

Often the natural sediment concentration profile will extend higher into the water column than the upper current meter. In `bblt` the parameter h_{max} governs the maximum height of the concentration profile. Clearly h_{max} must be less than the height of the upper current meter, since there is no velocity information above this height. The user can either accept this limitation or create synthetic time series higher in the water column. The technique of generating a synthetic time series by duplicating the upper current meter at a higher level (e.g. creating a time series at 20 m above the bottom which is identical to the observed currents at 10 m) creates a no shear zone which has an interesting side effect. In most cases the mean drift will be increased, but the mean diffusivity will be reduced because the mean shear in the water column has been reduced.

3 Input Parameters

The input parameters describing the physics of a `bblt` simulation were discussed in Section 2 and are listed in Table 1. A detailed description is provided here. A list

Table 1: List of input parameters for `bblt` v3.1.

$u(z,t)$, $v(z,t)$	the velocity field
w_s	sediment particle settling velocity
h_{max}	maximum height of suspended sediment
h_{ref}	sediment reference height
$a_0, a_1, \ldots, a_n$	bottom friction parameters ($C_d \equiv a_2$)
$\kappa = 0.4$	von Karman constant
N	number of packets
M	total mass of packets
t_m	vertical shuffling time scale
Δt_a	advection time step
Δt_U	time between updates of the velocity field
t_s	start time
t_t	total time

of additional useful symbols is given in Table 2. The relationship among the various time parameters is illustrated in Figure 5. In order to establish a connection between the general material presented here and the specific example given later, this section contains forward references to some of the tables in Section 6,

Velocity field Concurrent time series of easterly (u) and northerly (v) velocity at several vertical positions in the water column are required; ideally at least one vertical position should be within the bottom boundary layer. It is very important that the velocity time series are specified at the same times and that the time interval is constant (Δt_U). For example you cannot have one time series specified on the hours and another on the half hours. Once `bblt v3.1` finds the beginning of the time series (as determined by the start time t_s) it does not check the time stamps again.

As mentioned earlier, the source of the velocity data does not matter, it can be from current meters or numerical models. We refer to current meters for convenience. As mentioned later in the discussion of the source code, `bblt v3.1` can be configured to read harmonic constituents rather than time series, and the required input module is provided.

Velocity update time interval Δt_U is the time interval of the specified velocity time series. When the velocity is specified by harmonic components, then Δt_U determines how often the synthesis of harmonic components is done.

Particle settling velocity The settling velocity, w_s, should match that of the suspended sediment. We assume that the drill mud fines flocculate and form aggregates (flocs) with settling velocities much larger than the individual grains. Typical floc settling velocities are in the range of $0.1 - 1$ cm/s. The predicted dispersion and

Table 2: List of symbols

$u_* = \sqrt{\tau_b/\rho}$	friction (shear) velocity
ρ	water density
$u_*^2 = C_d\ \mathbf{u}^2$	typical estimate of u_*
$\mathbf{u}$	magnitude of observed near bottom current
$p = w_s/\kappa u_*$	Rouse number
$c(z) \equiv c_h(h/z)^p$	the Rouse concentration profile
$\hat{c}(z)$	normalized concentration profile ($\int_{href}^{hmax} \hat{c}(z)\ \mathrm{d}z \equiv 1$)
$F(z)$	fraction of the mass in the interval $h_{ref} < z' < z$
α_n	rank which maps $F(z) \rightarrow Z_n$
$X_n(t)$, $Y_n(t)$	horizontal position of the n^{th} packet at time t
$Z_n(\mathrm{t})$	vertical position of the n^{th} packet at time t
n	packet label, $n = 1, 2, \ldots, N$
t	time
z	vertical coordinate, $z = 0$ at the sea floor and positive upward
$q = \Delta t_U/\Delta t_a$	an integer, see algorithm outline
$k = N\Delta t_a/t_m$	the number of vertical shuffles performed each Δt_a
steps $= t_t/\Delta t_U$	number of velocity field updates

transport of sediment are strongly dependent on w_s and in circumstances where it cannot be well defined a sensitivity analysis should be performed.

Maximum height of sediment profile Typically the bottom boundary layer is restricted to a few 10's of meters above the sea floor. Often the bottom boundary layer is restricted by stratification in the water column. The user must specify the maximum height (h_{max}) of the concentration profile. `bblt` sets h_{max} to the minimum of the user specified h_{max} and the height of the upper current time series.

Sediment reference height The sediment reference height (h_{ref}) is a mathematical construct which defines the lower boundary of the suspended sediment load in the Rouse profile. h_{ref} must not be less than the roughness height (ξ_0) used in the log-layer calculation (Appendix A) and a typical value for `bblt` applications has been 0.35 cm.

Bottom friction parameters The friction velocity u_* is computed from the near-bottom currents using the polynomial stress law (2). The typical form used is $u_*^2 = C_d|\mathbf{u}_b|^2$ where C_d is a drag coefficient whose value is taken as 0.005 at 1 meter above the bottom. The constant a_0 may be used to represent background stress associated with waves. Notice that `bblt` calculates a drag coefficient separately for extrapolation of the currents below the bottom current meter record (Appendix A).

Number and mass of packets The mass of material to be modelled, M, is divided into N equal packets. Because output concentrations are calculated by counting the

number of packets within a specified sampling volume v, the minimum detectable concentration $M/(Nv)$.

Vertical shuffling time scale The vertical shuffling time scale, t_m, is the time required to do N pair-wise packet exchanges. The shuffling time scale is proportional to the vertical mixing time scale (average time for a particle to sample the entire vertical profile; Hannah et al., 1996). When t_m is set to 3 hours, dispersion is maximized for semi-diurnal forcing. In realistic simulations, the net dispersion changes by roughly a factor of 2 for t_m in the range of 1 to 10 hours.

Advection time step The advection time step, Δt_a, is the length of time each particle is advected before the vertical shuffling takes places. At the end of each advection time step, $k = Nt_a/\Delta t_m$ pair-wise packet shuffles are performed. The input velocity time series (with time interval Δt_U) are interpolated to the advection time step and `bblt` requires $\Delta t_a \leq \Delta t_U$. In addition, it requires that $q = \Delta t_U/\Delta t_a$ be an integer (Table 3). `bblt v3.1` adjusts the user specified Δt_a to achieve this, and it requires that at least 1 vertical shuffle takes place each Δt_a. The program stops if $k < 1$.

Hannah et al. (1995) showed that dispersion is independent of Δt_a for $t_m > 2\Delta t_a$. For $t_m < 2\Delta t_a$ the granularity of the advection and the mixing becomes evident. The advection time step controls run time and $\Delta t_a = 0.1$ hrs is used in the sample run.

Start time The start time, t_s, is used as a pointer into the specified velocity time series. Each time series is searched until a record is found with time greater than or equal to $t_s - \Delta t_U/2$. The next record is the first one used and defines the start time. The first and second velocity record for each time series are written to the echo file and should be checked.

When the velocity is computed from harmonic constituents then t_s can be used to adjust the phase.

Total time The total integration time, t_t, controls the length of the simulation.

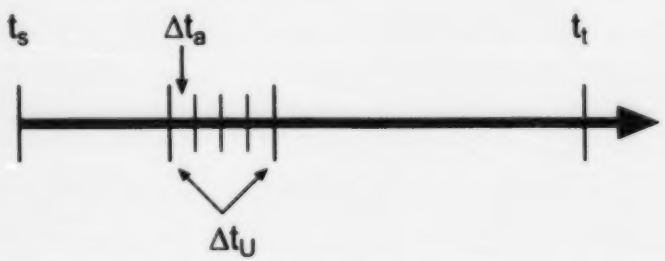

Figure 5: Schematic time line showing the relationship among t_s, t_t, Δt_a, and Δt_U.

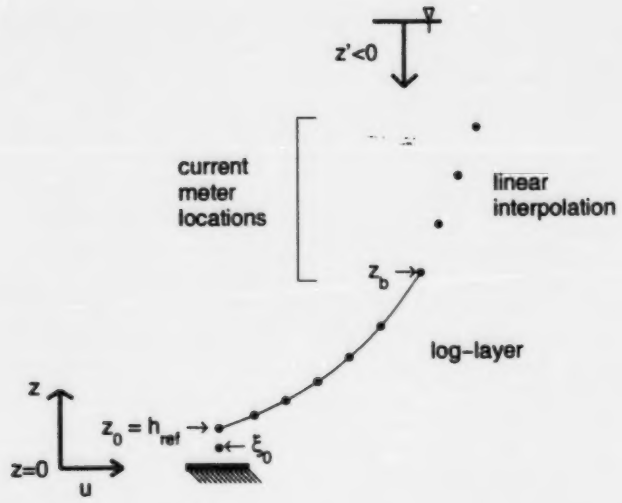

Figure 6: The water column as seen by `bblt`. The vertical coordinate is schematic and not to scale.

Another set of inputs is required to describe the vertical locations where the velocity field will be specified and/or calculated. Here we give some background on the requirements. A detailed example and explanation is provided for the sample run in Section 6.

The user must specify the currents at several vertical locations in the water column. While `bblt` works in terms of distance above the bottom, the vertical positions of the current meters are specified with respect to the surface of the ocean. The locations are described in a coordinate system (z') that is zero at the surface and positive upwards so the current meter locations are generally negative (Figure 6 and Table 7). The connection between the specified vertical locations and the bottom of the ocean is provided as part of the specification of the near-bottom log layer.

The height of the bottom current meter above the bottom of the ocean comes from the information the user gives as part of the calculation of the velocity profile below the bottom current meter (the log layer described in Appendix A). The calculation requires that the user specifies the vertical locations where this profile will be calculated (e.g. Table 6). The top of this profile specifies the height of the bottom current meter above the bottom of the ocean (z_b), which together with the depth of the current meters below the surface gives the total water depth and the height of all the current meters above the sea floor (Figure 6). The total water depth is part of the standard output. `bblt` assumes a constant depth ocean and does not account for changing depth due to processes such as waves.

The bottom of the log layer, z_0, (Figure 6) specifies the minimum height above the bottom where the currents are calculated. `bblt v3.1` requires that $z_0 = h_{ref}$. A third near-bottom length scale is the roughness length ξ_0 from the log-layer calculation, `bblt v3.1` requires $\xi_0 < h_{ref}$. The program stops if these conditions are not satisfied.

Running `bblt` requires other parameters. For example a seed is required for the random number generator, a number is required to specify the method used for vertical shuffling and there are many input parameters required to control the output. These are described in Section 6 as part of the sample run.

4 Model Output and Derived Quantities

The model outputs are the time series of positions of the packets

$$X_n(t),\ Y_n(t),\ Z_n(t) \quad \text{for} \quad n = 1, \ldots, N \tag{6}$$

where N is the number of packets, $X_n(t)$ is the position along the x coordinate of packet n at time t, Y is the y coordinate and Z is the vertical coordinate (positive upwards from the sea floor). Everything else is a derived quantity. Here we define the basic derived quantities.

First we define the center of mass of the cloud of packets at time t $(\overline{X}(t), \overline{Y}(t), \overline{Z}(t))$ where

$$\overline{X}(t) = (1/N) \sum_{n=1}^{N} X_n(t) \tag{7}$$

and the other two components are defined in the same manner. We also define z_{min}, z_{max}, z_{cm} which are the minimum, maximum, and average of $\overline{Z}$ over the simulation.

The mean drift (U, V) is the time rate of change of the horizontal center of mass over the simulation. U (V) is computed as the slope of a straight line fit to the time series $\overline{X}(t)$ $(\overline{Y}(t))$.

The horizontal spreading of the patch about its center of mass is measured by the horizontal variance of the packet distribution σ^2. In one dimension the variance of N packets is

$$\sigma^2(t) = \frac{1}{N-1} \sum_{n=1}^{N} (X_n - \overline{X})^2 \tag{8}$$

In two dimensions we calculate σ^2_{max} and σ^2_{min} which are the variances along the major and minor axes of the horizontal projection of the distribution (i.e. the vertical coordinate is ignored). At each time, σ^2_{max} and σ^2_{min} are the eigenvalues of the covariance matrix

$$C = \begin{bmatrix} \sigma^2_{xx} & \sigma^2_{xy} \\ \sigma^2_{xy} & \sigma^2_{yy} \end{bmatrix}$$

where

$$\sigma_{xx}^2 = \frac{1}{N-1}\sum_{n=1}^{N}(X_n - \overline{X})^2$$

$$\sigma_{yy}^2 = \frac{1}{N-1}\sum_{n=1}^{N}(Y_n - \overline{Y})^2$$

$$\sigma_{xy}^2 = \frac{1}{N-1}\sum_{n=1}^{N}(X_n - \overline{X})(Y_n - \overline{Y})$$

The solutions are

$$\sigma_{max}^2 = (\sigma_{xx}^2 + \sigma_{yy}^2) + A$$
$$\sigma_{min}^2 = (\sigma_{xx}^2 + \sigma_{yy}^2) - A$$

where

$$A = [(\sigma_{xx}^2 - \sigma_{yy}^2)^2 + 4(\sigma_{xy}^2)^2]^{1/2}$$

(Preisendorfer, 1988).

The effective horizontal diffusivities along the major (K_{max}) and minor (K_{min}) axes are then defined as

$$K_{max} = \frac{1}{2}\frac{\partial < \sigma^2{}_{max} >}{\partial t} \quad ; \quad K_{min} = \frac{1}{2}\frac{\partial < \sigma^2{}_{min} >}{\partial t} \tag{9}$$

(e.g. Csanady 1973) where $< \cdot >$ denotes a time averaging which smoothes out short-term fluctuations such as the tides.

The interpretation of the mean drift and the horizontal diffusivity is straight-forward for bulk releases. These quantities are less meaningful for continuous scenarios.

The calculation of concentration (mass per unit volume) is straight-forward. The number of packets in the user specified sampling volume are counted and the concentration is calculated using the known volume and packet mass. The sampling volume is typically a cylinder of radius r extending in the vertical between z_1 and z_2. A standard `bblt` output is the near-bottom concentration following the center of mass of the sediment distribution.

5 Software Overview

In this section, the `bblt v3.1` code is briefly described. This description should help the user understand how `bblt` functions and aid in making modifications. The user is expected to modify the output modules (`Output`) and the modules that describe the interaction of the sediment with the bottom: the critical threshold stress (`Threshold`) and the fraction of the sediment in suspension (`SuspendedFraction`). The user may also wish to modify the way the input is done.

The computer code is written in FORTRAN77 and is reasonably portable; for example it has been successfully compiled and run using the fortran to c converter f2c. A complete list of source code files is given in Appendix B.

5.1 bblt v3.1

The program is organized as a main module which implements the algorithm shown in Table 3 and submodules (functions) which perform specific tasks. The modules are listed below in the order they are executed in the main loop and a brief description is given. Where the functional name is different than the file name, the functional name is bracketed.

bbltv31.f This is the main module for **bblt**. Here, most of the variables are initialized, time stepping is organized and error reporting is carried out. The program listing for the main module is given in Appendix C

This file also contains **ErrorReport**, a module for writing error messages to both standard output and the echo file.

UserSetup.f A place for the user to open files and do other setup activity. It is called early in the initialization process. At present this module is empty.

RdParams2.f (RdParams) Reads the input parameters, writes them to standard output, and converts input parameters to common units of time (hours).

nodeinfo2.f (NodeInfo) The vertical location of the current observations is specified by the user in terms of distance below the surface of the ocean, however **bblt** works in terms of height above the bottom. Based on the information about the vertical location of the current observations and the user supplied information about where to evaluate the velocities in the bottom log layer, this subprogram constructs a list of heights (or nodes) above the bottom where velocities will be specified. A sample input file is shown in Table 6.

velinfo5.f (VelInfo) Updates the velocity field at each Δt_U and computes the velocity profile below the bottom velocity observation. In this module, the velocity components are read from user specified files.

The calculation of the near-bottom velocity profile is based on the logarithmic profile described in Appendix A. The calculation assumes a quadratic stress law and the free parameter ξ_0 (roughness length) is hard-coded into the subroutine. The calculation of the logarithmic profile is separate from the calculation of the friction velocity (u_*) used to calculate the sediment concentration profile. In particular, changing the coefficients in the polynomial stress law (2) has no effect on the log-layer. Changes to the log-layer are made by changing the code in **velinfo5.f**.

Table 3: Outline of the algorithm for `bblt v3.1`. The symbols are defined in the text and in Tables 1 and 2.

Define:

$q = \Delta t_U / \Delta t_a$ number of distinct advection steps (and mixing events) between updates of the velocity field.
Δt_a is adjusted so that q is an integer

Initialize:

specify input parameters: t_m, Δt_a, Δt_U, t_s, t_t, w_s, h_{ref}, κ, C_D, h_{max}
$t = t_s$
create packets: specify N, $X_n(t_s)$
assign the α_n
get initial velocity field: $\mathbf{v}(t_s)$
output

Do Sediment Transport:

do until time $t \geq t_s + t_t$
get $\mathbf{v}(t + \Delta t_U)$
do j = 1, q
compute $\mathbf{v}(z, t)$
compute parameters required for concentration profile
compute the probability distribution $F(z)$
perform k pair-wise exchanges of the α_n
compute vertical positions of packets: Z_n
advect packets in the horizontal: X_n
enddo
$t = t + \Delta t_U$
output
enddo

`velinfo3.f (VelInfo)` Performs the same function as **`velinfo5.f`** except the velocities are constructed from user specified harmonic components, rather than time series. This module can be used in place of **`velinfo5.f`**.

`InterpVel.f` Linearly interpolates velocity data from the velocity update time steps (Δt_U) to the advection time steps (Δt_a).

`CalcUstar.f` Uses the velocity and specified friction parameters to calculate the friction velocity (u_*). This function could be modified to read a time series of friction

velocity. Some care is required to make sure that the time stamps on the stress and velocity time series are correctly aligned.

SedFcns.f A collection of functions related to the suspended sediment.

Rousenum Calculates the Rouse number p based on u_*, κ and w_s.

Threshold User defined critical shear velocity, u_{*crit}.

SuspendedFraction User defined function which calculates the fraction of sediment in suspension based on u_* and u_{*crit}.

CrtPckts2.f Initializes packets and assigns the initial α_n used for the vertical shuffling. It also adds packets if required for a continuous release scenario.

VertShffl.f (VertShuffl) Does the vertical shuffling. The options are no shuffling and pairwise exchanges of randomly selected packets.

PDFcompute.f Converts the updated Rouse profile into a table of cumulative probability at logarithmically spaced depths (smaller intervals near the bottom). The number of values in the table is controlled by the variable **nentry** in this subroutine. The current value is **nentry=50**.

AbRVertPos.f Adjusts the vertical position of the packets to fit the updated probability distribution. Based on the details of **PDFcompute**, material not in suspension is assigned a height $z = h_{ref}/2$. This allows the user to choose whether the material not in suspension is counted as part of the near-bottom concentration. The choice is made via the choice of the lower limit in the concentration calculation.

VelHoriz.f Calculates the velocity at the vertical position of a given packet using linear interpolation.

MoveHoriz.f Moves all the particles in the horizontal.

output4b.f This module contains the subroutines listed below which initialize the standard output files and calculate and record the results at user specified intervals.

output Organize the calculation of the output.

sedout Packet information

statout Standard statistics

velout Velocity information

outfileopen Open an output file

packout Packet positions

controut Time series of concentration at specified locations

contrmaout Concentration maps

`Cncentr1` Calculate concentration by packet counting in 3-d.

`Cncentr2` Calculate concentration by packet counting in 2-d and using the Rouse profile.

`ran1.f` The random number generator used with `bblt v3.1` was ran1.f from Numerical Recipes (Press et al., 1992). We cannot distribute it because of copyright restrictions. The user must supply a similar random number generator.

5.2 Extensions of Local `bblt` since Version 1.0

Five important extensions have been developed since `bblt` v1 (the version described in Hannah et al. 1995, 1996):

Modified Bottom Stress Law. The linear bottom stress law in `bblt` v1 has been replaced by the polynomial stress law (2) which allows for much more flexibility in specifying u_*. This item was discussed in Section 2.

Logarithmic Bottom Layer. The linear vertical structure of the bottom sublayer ($z < \xi_b$) in `bblt` v1 has been replaced by a logarithmic bottom sublayer. The implementation of the logarithmic sublayer follows the formalism developed for the finite-element circulation model QUODDY which has been used to generate velocity profiles used in several applications. Thus, the logarithmic sublayer used in the `bblt` models is consistent with the velocity profiles taken from the circulation model. It is also a reasonable approximation for use with the observed currents from levels within a few metres of the seafloor (within the constant-stress bottom region), but a cruder approximation for observed currents from greater distances above the seafloor. Details regarding the implementation of the logarithmic layer are provided in Appendix A.

Maximum Sediment Profile Height. In `bblt` v1, the maximum height of the concentration profile was set by the height of the upper current data level (e.g. the upper current meter). Now the user can specify the maximum height (h_{max}) of the concentration profile which must be less than or equal to the height of the upper current data level. This feature is particularly important in areas where stratification limits the height of the bottom mixed layer (beyond that implied by the sediment height in an unrestricted Rouse profile).

Concentration Calculation. The `bblt` models compute time series of concentration at user-specified locations. The concentration calculation is a straight-forward exercise of packet counting in specified cylinders centered on the locations of interest, followed by multiplication by the packet mass and division by the cylinder volume.

Continuous Release. `bblt` v1 was limited to bulk release scenarios in which there is no mass input beyond the initial distribution. `bblt v3.1` can handle simple continuous release scenarios. The continuous release capability is a straight-forward extension of the bulk release method. Let N be the maximum number of packets and let $n = 1, \ldots, N$ be the packet identifier. All N packets are assigned an α_n drawn from a uniform random distribution in the interval (0,1). At time t_1 only the N_1 packets in the interval $1, \ldots, N_1$ are involved in the advection and shuffling. When Δn packets are added at time t_2 then the packets in the interval $1, \ldots, N_1 + \Delta n$ are involved in the advection and shuffling. At all times the active α_n represent a realization of a uniform random distribution between (0,1).

There have also been modifications to `bblt` which have been reported but are not included in `bblt v3.1`. These include the ability to specify a time series of packet input and hence increase the complexity in the drilling waste discharge scenarios (e.g. Loder et al., 2001) and the ability to read a time series of bottom stress which allows for cases where the bottom stress is not solely a function of the currents (e.g. wave influences). These will be described in a future report.

6 Sample `bblt` run

In this section, we describe how to run `bblt v3.1`. It can be run at the command line, however given the large amount of input, an input file should be set up with all of the information that the program requires. The code provides no defaults for omitted inputs.

6.1 Pseudocode Input

Before describing the sample input file, an input file is presented in pseudocode (Table 4) and described line by line. The sample input file is presented in Table 5 in the next subsection.

Line 1 All of the start-up information that appears on the screen during the run is also sent to a file, sometimes called an 'echo' file. If the file is to be saved in a directory other than the one in which the program is run, then the path should be specified as well.

Line 2 Packets are shuffled randomly at each time step so a number is required to seed the random number generator. The number used should be on the order of 2^{32}, or 8 or 9 digits long. Note that a given seed will always produce the same sequence of random numbers, so two model runs with the same input will produce identical output.

```
1     echo_filename
2     random_number_seed
3     total_time(hr)    start_time(hr)
4     time_between_vel_updates(hr)
5     dta(hr) dtms(hr)
6     vmethod
7     number_of_friction_parameters
8     a0 a1 ... a[number_of_friction_parameters]
9     settling_velocity(m/s)
10    kappa reference_height(m)
11    maximum_height (m)
12    total_mass(g) total_packets start_release(hr) end_release(hr)
13    log_layer_filename
14    current_height_filename
15    current_record_filename1 **
16    time_step_for_sediment_parameters_file(s)
17    sediment_parameters_filename
18    time_step_for_velocity_file(s)
19    velocity_filename
20    time_step_for_statistics_file(s)
21    statistics_filename
22    bottom_of_cylinder(m) top_of_cylinder(m) radius_of_cylinder(m)
23    time_step_for_packet_positions_file(s)
24    packet_positions_filename
25    time_step_for_concentration_file(s)
26    concentration_filename
27    number_of_positions_at_which_to_calculate_concentration
28    x(m) y(m) radius(m) bottom_of_cylinder(m) top_of_cylinder(m) **
29    time_step_for_concentration_map_file(s)
30    concentration_map_filename
31    xgrid xmin(m) xmax(m) ygrid ymin(m) ymax(m)
         radius_of_cylinder(m) reference_height(m) top_of_cylinder(m)
```

Table 4: Pseudocode representation of the bblt input file. Lines marked with ** may represent multiple lines, depending on the number of current records, or the number of positions at which to calculate concentration.

Line 3 The duration of the run `total_time` (t_t) and the start time of the run `start_time` (t_s) are specified here. Units must be paid very close attention. The time stamp in the velocity file is in days, but the start time for the simulation is given in hours relative to day zero. Other time-related input parameters are also given in hours, but the interval at which to output statistics must be specified in seconds.

Line 4 The time between velocity updates (Δt_U) is the interval between velocity measurements in the velocity time series. It must be given in hours.

Line 5 The advection time step `dta` (Δt_a) and vertical shuffling time scale `dtms` (t_m) are specified in hours.

Line 6 `vmethod` is the number of the scheme to use for vertical shuffling. If `vmethod` = 1, then there is no shuffling. If `vmethod` = 2, then the packets are shuffled k times every advection time step, which is the normal setting.

Line 7 The number given here tells `bblt` how many friction parameters will follow on line 8. If there are three friction parameters, then the friction velocity calculation is defined by a 2nd order polynomial, see (2).

Line 8 Here the friction parameters (a_i) are given. They are listed from zeroth to $(n-1)$th order where n is the number of friction parameters (Line 7).

Line 9 The settling velocity of the packets (w_s) is given in m/s.

Line 10 This line must contain two inputs: the von Karman constant $\kappa = 0.4$ and the reference height, h_{ref} (metres). All lengths in `bblt` have units of metres.

Line 11 The maximum height of sediment (h_{max}) is positive, and given in meters above the bottom.

Line 12 Four inputs are specified on this line. First is the total mass of material, M, in units of grams, and then the total number of packets, N. The time to begin releasing packets is given in hours relative to the beginning of the run (time $t = t_s$). If all of the packets are to be released at the beginning of the run, then the start of the release and the end of the release must be equal. If the end of the release time is specified as some other time, packets will be introduced at a constant rate at each advection step (Δt_a in Line 5).

Line 13 This line gives the name of the file containing the heights in the log layer where velocities are to be calculated. A sample version is given in Table 6. The first two lines are header lines, followed by the number of heights in the file. The heights themselves are roughly logarithmically spaced with more points closer to the bottom. The heights must range from h_{ref} to the height of the bottom current measurement, which in Table 6 is 3 mab (meters above the bottom). Since linear

interpolation is used to calculate the velocities between the specified locations, high resolution is required to approximate a log-layer. Note that if there are only two heights, h_{ref} and the height of the bottom current meter, a linear boundary layer will result.

Line 14 This contains the depths of the current meters. Table 7 shows an example of this file. The first two lines are header lines. The third gives the number of depths to be read. The fourth and fifth are hold-overs from another type of standard input file but have no importance here. The subsequent lines are the depths (written in the second column). The ordering of the depths is important. The depths must be ordered from the bottom up. The coordinate system is positive upwards with zero at the sea surface, so the depths are negative. The first column must be present but has no meaning.

Line 15 In the lines after line 14, the files containing the velocity information are specified. The number of lines depends on the number of records. The number and order must correspond with the depths given in the file in Line 14.

A sample velocity file is given in Table 8. `bblt` allows a header of any length to be included at the beginning of the file, but it must be followed by a line containing `XXXX`. The next four lines must contain two header lines, a line containing the year, and then a line with the number of columns to be read in (normally 3). Then follows the velocity time series, with the time in year-days, and the $u-$ and $v-$ components of velocity in m/s. It's important to make sure that there are no gaps in the data, and that the interval between data points is constant (and equal to Δt_V).

Line 16 The rest of the lines relate to model output. This line is the time step at which to send output to the sediment parameters file. If the time given is equal to one, then output is sent to the file every Δt_a. If the time is less than or equal to zero, then no output is written. Otherwise, output is written at the time step indicated (in seconds). These rules apply for all the output files below.

If the option with the output time step ≤ 0 is used then the other auxilary information, such as the file name, is not asked for and must not appear in the input stream.

Line 17 The name of the file in which to store sediment information, `sediment_parameters_filename` in Table 4. The file format is defined by the module `sedout`. The details are given in Section 6.3 and Table 9.

Line 18 The time step (seconds) at which to output information about the velocity field.

Line 19 The name of the file in which to store velocity information, **velocity_filename** in Table 4. The file format is defined by the module **velout**. The details are given in Section 6.3.

Line 20 The time step for statistics output.

Line 21 The name of the statistics file, **statistics_filename**. The file format is defined by the module **statout2**. Sample output is reported in Table 10 and details are given in Section 6.3.

Line 22 One of the statistics reported is the concentration of sediment at the horizontal centre of mass of the sediment patch ($\overline{X}, \overline{Y}$, see Section 4). To calculate the concentration, **bblt** counts the number of packets in a cylinder centred at the centre of mass. This line gives the height (above the bottom) of the bottom and top of the cylinder and the radius, all in metres. The bottom of the cylinder should be at the reference height. This is good for computing the near bottom concentration of the suspended fraction, which is typically of concern. Other choices are possible.

Line 23 The time step in seconds at which to output the packet positions ($X_n(t)$, $Y_n(t)$, $Z_n(t)$).

Line 24 The name of the file in which to store the packet position information (**packet_positions_filename**). The file format is defined by the module **packout**. Further details are given in Section 6.3 and Table 11.

Line 25 **bblt** records concentration time series at locations specified by the user. This line tells it how often to save the concentrations

Line 26 The name of the file in which to store the concentration time series (**concentration_filename**). The file format is defined by the module **controut**. Further details are given in Section 6.3.

Line 27 The number of locations at which to record concentration time series.

Line 28 The locations and sampling volumes are listed in the lines following Line 27. There must be as many locations and sampling volumes as specified above. The first two numbers are the easterly- and northerly- coordinates relative to the release point [0,0], and the third is the radius of the cylinder. The last two numbers are the bottom and top of the cylinder. These are all given in meters.

To calculate the near-bottom concentration of the suspended sediment, the bottom of the cylinder should be set to h_{ref}. However, if **bblt** is configured to allow for some material to be out of suspension (using **Threshold** and **SuspendedFraction**) then this material can be included in the near-bottom concentration by setting the bottom of the cylinder to a number $< h_{ref}/2$ (see **AbRVertPos.f**). This capability has not been widely explored.

Line 29 The final part of the input file relates to concentration maps. Snapshots of concentration are recorded at the interval specified here for the grid defined in Line 31.

Line 30 The output file name for concentration maps. The file format is defined by the module `contrmaout`. Further details are given in Section 6.3.

Line 31 The grid is defined on this line. The order is as follows: the number of grid points in the easterly- direction followed by the minimum and maximum values, the number of grid points in the northerly- direction followed by the minimum and maximum values, the radius of the sampling cylinder, and the bottom and top of the cylinder. All lengths are in meters.

In its present configuration `contrmaout` insists that the sampling cylinder radius be less than or equal to 1/2 the distance between the grid points. The final sampling radius is reported in the model output. `contrmaout` should be modified to satisfy the user's requirements.

6.2 Input

As an example, we present results from a `bblt` run. The input file is shown in Table 5. The auxiliary files called by `bblt` in lines 13 and 14 are shown in Tables 6 and 8. This run calls in 4 velocity records and an example of one of them is shown in Table 7. The line numbers are included for easy reference, and must not be present when `bblt` is run.

```
1     echo.txt
2     948321757
3     125  4534.0
4     0.5
5     0.1  4
6     2
7     3
8     0.  0.  0.0035
9     0.005
10    0.4  0.0035
11    34.0
12    94800000  100000 1 1
13    hloglayer2.3m
14    GBFS2.heights
15a   GBFS2_64m.old
15b   GBFS2_57m.old
15c   GBFS2_35m.old
15d   GBFS2_11m.old
16    1
17    sedparams.out
18    3740
19    velocity.out
20    3740
21    stats.out
22    0.0035 0.1035 500
23    3740
24    pack.out
25    374
26    cnstr.out
27    1
28a   0.0     0.0      500 0.0035 0.1035
29    44880
30    cnstrmp.out
31    31 -5000 50000 31 -50000 5000 1000 0.0035 0.1035
```

Table 5: Sample `bblt` input file. This particular simulation was for a site on Georges Bank, beginning on July 8 (hour 4534.0 since Jan 1), and lasting a total of 125 hours (5 days). The column of line numbers is for reference only and must not be present in an actual `bblt` run.

```
# loglayer heights for linear shear in bottom 3 m.
#
 25
 0.0035
 0.0040
 0.005
 0.006
 0.007
 0.008
 0.01
 0.012
 0.015
 0.018
 0.02
 0.03
 0.04
 0.05
 0.06
 0.07
 0.08
 0.1
 0.2
 0.3
 0.5
 0.7
 1.0
 2.0
 3.0
```

Table 6: Sample 'log_layer' file (Line 13 in Table 4). For this case, the bottom current meter is 3 mab, so the heights in the log layer are defined between the reference height (0.0035 m) and 3 m. They are approximately logarithmically spaced. The first two lines are header lines, followed by the number of depths in the file.

```
-STATION INFORMATION - one line with:
 latitude (positive North), longitude (positive W), sounding depth (m),
 instrument depth (m), station number, site number, start (year,
 year-day, decimal hours), end (year, year-day, decimal hours),
 sampling interval (decimal hours), number of cycles.

42.0803 66.8012 67.0 64.0 891 2 1988 179 19.997 1988 193 14.49750 .50 662
  3 U(891.,  64.M.)      M/SEC          64.00   -.24111
  4 V(891.,  64.M.)      M/SEC          64.00   -.24250
  7 T(891.,  64.M.)      DEG.C.         64.00   0.00000
  0                                      0.00   0.00000
  0                                      0.00   0.00000
XXXX
Georges Bank Frontal Study 2 site 891, 64 m, sampling interval 0.5 hr
time (utc0), u(m/s), v(m/s)
1988
3
178.8332        -0.2200 -0.0700
178.8541        -0.3000 0.1000
178.8750        -0.2800 0.2604
178.8959        -0.2802 0.3401
178.9168        -0.3000 0.3505
178.9377        -0.2987 0.4004
178.9586        -0.2083 0.4289
178.9795        -0.1183 0.3677
179.0004        -0.0283 0.2469
179.0213        0.0403  0.1171
179.0422        0.0506  0.0186
```

Table 7: Beginning of an observed currents file. (Line 15 in Table 4). Lines before XXXX are ignored. After comes two header lines, followed by year of observations, and number of columns in data file. The first column is the time in year, the second is the $u-$component of velocity and the third is the $v-$component. The $u-$component is east-west (positive east), and the $v-$component is north-south (positive north).

```
Georges Bank Frontal Study, site 2
distance below the surface
4
1
0.0
1   -64
1   -57
1   -35
1   -11
```

Table 8: Sample 'current_heights' file (Line 14 in Table 4). The first two lines are header lines, followed by the number of depths below the surface, two more space keeping lines, and then the depths of the current observations in the second column.

6.3 Output

In this section we take a detailed look at the output produced by a `bblt v3.1` run using the files in Tables 5-7 as input. The line numbers give below refer to the lines numbers in Tables 4 and 5.

Sediment parameters file (Line 17)

The sediment parameters file contains information about the calculated bottom shear velocity and the Rouse profile (Table 9). The columns are time (hours), u_*, $p = w_s/\kappa u_*$, and the number of active packets. The module that generates this file is `sedout`.

Notice that the first line (Table 9) has a a time stamp of zero. This line is generated during the the initialization process and must be removed before any time series analysis is done on this file. This feature is common to many of the `bblt v3.1` time series output files.

Velocity file (Line 19)

This file can contain the u and v components of velocity at each height above the bottom (those at which data are provided as well as calculated values in the near-bottom log layer). Generally, the columns are time (hours), height (m), u (m/s), and v (m/s). The module that generates this file is `velout`. The user should modify this module so that it writes the desired information.

0.	0.	0.	100000
4534.10	1.12693E-02	1.10921	100000
4534.20	1.13654E-02	1.09983	100000
4534.30	1.16232E-02	1.07544	100000
4534.40	1.20324E-02	1.03886	100000
4534.50	1.25782E-02	0.993784	100000
4534.60	1.31416E-02	0.951180	100000
4534.70	1.37627E-02	0.908255	100000
4534.80	1.45404E-02	0.859674	100000
4534.90	1.54511E-02	0.809003	100000

Table 9: First 10 lines of **sedparams.out**. The columns are time (hours), u_*, $p = w_s/\kappa u_*$, and the number of active packets.

Statistics file (Line 21)

This file summarizes the overall properties of the sediment patch (Table 10). The columns are: time, $\overline{X}$, $\overline{Y}$, $\overline{Z}$, σ^2_{max}, σ^2_{min}, θ, $c1$, $c2$, where time is in seconds, θ is the angle that the axis of maximum variance makes with the x-axis (positive = counter clockwise), $c1$ is the depth-integrated or 2-d concentration (g/m^2) at the center of mass (g/m^2), and $c2$ is the 3-d concentration (g/m^3) at the center of mass for the user defined cylinder radius and vertical extent. The other symbols are defined in Table 2. The module that generates this file is **statout2**.

Notice that the first line (Table 10) is a line of mostly zeroes which is generated during the initialization process. This line must be removed before any time series analysis is done on this file.

Packet positions file (Line 24)

This file contains the 3-D location of each packet at the time step specified by the user (Table 11). At each time step, **bblt** prints a line like:

```
ktotal, time   100000    4535.10    1.63264E+07
```

The first number (**ktotal**, or N) is the total number of packets, the second is the time in hours, and the third is the time in seconds (same reference). The next N lines contain the x, y, and z positions of each packet (all in meters) as well as the α_n. At the next time step, another line similar to the above will be printed, followed by the list of packets. The module that generates this file is **packout**.

```
0.00000000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.1207E+03 0.0000E+00
0.16326360E+08 0.7022E+03 0.7636E+02 0.8216E+01 0.2226E+06 0.1477E+05 0.3286E+02 0.9469E+02 0.4448E+02
0.16329960E+08 0.2024E+04 -.8328E+03 0.1283E+02 0.7323E+06 0.8563E+05 0.1580E+02 0.4762E+02 0.2185E+01
0.16333920E+08 0.3585E+04 -.3055E+04 0.1310E+02 0.1593E+07 0.3241E+06 -.4396E+01 0.1325E+02 0.1231E+01
0.16337520E+08 0.4856E+04 -.5528E+04 0.1374E+02 0.2540E+07 0.6707E+06 -.2599E+02 0.6188E+01 0.5070E+00
0.16341120E+08 0.5735E+04 -.8019E+04 0.1294E+02 0.3274E+07 0.9177E+06 -.4046E+02 0.5178E+01 0.9656E+00
0.16345080E+08 0.5963E+04 -.9655E+04 0.8056E+01 0.3945E+07 0.1159E+07 -.5084E+02 0.5099E+01 0.7037E+01
0.16348680E+08 0.5655E+04 -.1016E+05 0.3013E+01 0.4102E+07 0.1369E+07 -.5592E+02 0.5094E+01 0.2937E+02
0.16352640E+08 0.4817E+04 -.9653E+04 0.1064E+02 0.3924E+07 0.1510E+07 -.5505E+02 0.5350E+01 0.3428E+01
```

Table 10: First 10 lines of `stats.out`, the statistics file. The columns are time, $\overline{X}$, $\overline{Y}$, $\overline{Z}$, σ^2_{max}, σ^2_{min}, θ, $c1$, $c2$, where time is in seconds, θ and $c1$ are defined in the text and $c2$ is the 3-d concentration (g/m^3) at the center of mass. Notice that the first line is a line of mostly zeroes which is generated during the initialization process. This line must be removed before any time series analysis is done on this file.

ktotal, time	100000	4535.10	1.63264E+07
244.414	-22.846	0.028	0.061915
157.683	-42.192	0.044	0.081515
103.845	-29.066	0.021	0.050675
708.946	-60.207	1.399	0.352610
1148.646	372.096	23.649	0.892820
373.648	-66.926	0.331	0.206611
519.936	-71.577	0.861	0.296276
627.687	-111.935	2.982	0.457207
296.254	-70.411	0.217	0.174378

Table 11: First 10 lines of `pack.out`

Concentration	
x y z initial_concentration	
0. 0.	5.35000E-02 0.
0.000000E+00	0.0000E+00
0.453420E+04	0.5777E+03
0.453430E+04	0.5459E+03
0.453440E+04	0.4978E+03
0.453450E+04	0.4389E+03
0.453460E+04	0.3843E+03

Table 12: First 10 lines of `cnstr.out`. In this example there is only one time series location, (x,y) = (0,0).

Concentration file (Line 26)

This file (`cnstr.out`) contains the time series of concentration at locations specified by the user. After two header lines, `bblt` lists the positions of the time series locations in x, y, and $(z_{bottom} + z_{top})/2$ as well as the initial concentration. After a blank line follow columns of numbers representing the concentrations in g/m^3 (or mg/l). The first column is time in hours, and all others are concentrations. There are as many columns as user-specified locations. An example is given in Table 12. The module that generates this file is `controut`.

Concentration map file (Line 30)

The concentration map (or snapshot) file is quite long and is not shown. The module that generates this file is `contrmaout`. The basic structure is as follows. The first line is

a header line. The second line is the number of grid points in the x-direction (**xgrid** = 31 from Tables 4 and 5). The next **xgrid** lines are the x-coordinates of the grid points written out 1 per line. The next line is the number of grid points in the y-direction (**ygrid** = 31 from Tables 4 and 5). The next **ygrid** lines are the y-coordinates of the grid points written out 1 per line. That completes the definition of the grid on which the concentrations are computed. The rest of the file is the data. Each data block consists of a time stamp followed by the concentrations (in g/m^3) at each point on the grid, written 1 per line.

In its present configuration **contrmaout** insists that the sampling cylinder radius be less than or equal to 1/2 the distance between the grid points. The final sampling radius is reported in the model output. **contrmaout** should be modified to satisfy the user's requirements.

6.4 Plots

Figure 7 is a graphical representation of the output from the sample run described above. The upper left panel shows the near-bottom concentration following the center of mass (column 9 from **stats.out**) and at the release point (**cnstr.out**). The middle left panel shows the mean height of the vertical center of mass of the sediment distribution ($\overline{Z}(t)$ from column 4 of **stats.out**). The lower left panel shows the evolution of the horizontal variance of the distribution (σ^2_{max} and σ^2_{min} from columns 5 and 6 **stats.out**). The upper and middle panels on the right, show snapshots of the near bottom concentration taken from **cnstrmp.out**. The lower right panel shows the motion of the horizontal center of mass ($\overline{X}$, $\overline{Y}$ from **stats.out**).

7 Final Comments

bblt was designed as a framework for testing ideas about suspended sediment transport, and as a tool for estimating drift, dispersion and concentrations of suspended drill wastes with known inputs and physical forcing. Like all models, it will always have limitations and aspects which can be improved. The design assumes that

- users will configure the input and output modules to suit their needs;
- it should be possible to change the details of the concentration profile without recoding the whole model;
- users need to be able to access the details of the critical shear stress;
- users need to be able to change the rules for the critical shear stress and the rules that govern the fraction of sediment that is in suspension;

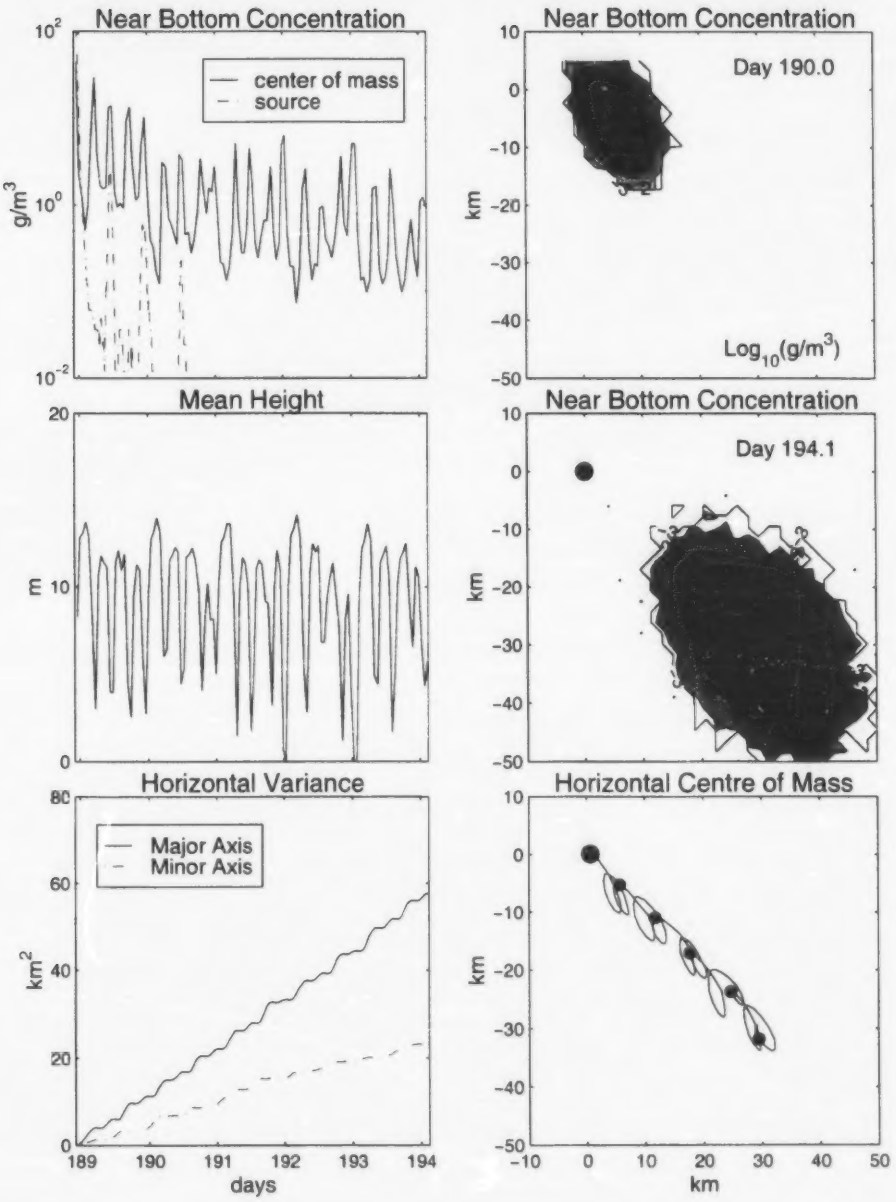

Figure 7: Graphical representation of output from example case. See the text for details.

- display and analysis of the results are external activities.

Consistent with this design philosophy, the code is modular and `bblt v3.1` performs only very simple analysis and does not have graphical capability. The output file structures are very simple and the output is easily incorporated into packages such as MATLAB.

In closing, a successful run of `bblt v3.1` results in the phrase 'Clean Exit' being written to standard output and the echo file. If this phrase is not written out then `bblt v3.1` did not run correctly.

Acknowledgements

We thank John Loder for ongoing encouragement and support in the development and application of `bblt` and Yingshuo Shen for contributions to model implementation. This work was supported by the (Canadian) Federal Program for Energy Research and Development.

References

Anderson, C. 2001. Seabed shear stress estimates at the Cohasset and Hibernia oil production sites. Can. Data Rep. of Hydrog. and Ocean Sci. (In prep.).

Boudreau, P. R., et al. 1999. The possible environmental impacts of petroleum exploration activities on the Georges Bank ecosystem. Can. Tech. Rep. Fish. Aquatic Sci. 2259. vi+106pp.

Csanady, G. T. *Turbulent Diffusion in the Environment.* D. Reidel Publishing Company, 1973.

H. B. Fischer, E. J. List, R. C. Y. Koh, J. Imberger, and N. H. Brooks. *Mixing in Inland and Coastal Waters.* Academic Press, 1979.

Gordon, D. C., P. J. Cranford, D. K. Muschenheim, J. W. Loder, P. D. Keizer and K. Kranck. 1993. Predicting the environmental impacts of drillings wastes on Georges Bank scallop populations. In P. M. Ryan, editor, *Managing the Environmental Impact of Offshore Oil Production*, pages 139–147. Canadian Society of Environmental Biologists.

Gordon Jr., D. C., P. J. Cranford, C. G. Hannah, J. W. Loder, T. M. Milligan, D. K. Muschenheim and Y. Shen. 2000. The potential effects of exploratory hydrocarbon drilling on Georges Bank scallop populations. Can. Tech. Rep. Fish. Aquat. Sci. 2317: 116 pp.

Hannah, C.G., Y. Shen, J.W. Loder and D.K. Muschenheim. 1995. `bblt`: Formulation and Exploratory Applications of a Benthic Boundary Layer Transport Model. Can. Tech. Rep. Hydrogr. Ocean Sci. 166: vi + 52 pp.

Hannah, C. G., J. W. Loder, and Y. Shen. 1996. Shear dispersion in the benthic boundary layer. In: Estuarine and Coastal Modeling: Proceedings of the 4th International Conference. M. L. Spaulding and R. T. Cheng eds. ASCE. Pages 454-465.

Hannah, C.G., Z. Xu, Y. Shen, and J. W. Loder. 1998. Models for suspended sediment dispersion and drift. In: Estuarine and Coastal Modeling: Proceedings of the 5th International Conference. ASCE. Pages 708-722.

Loder, J.W., C. G. Hannah, Y. Shen, E. A. Gonzalez, Z. Xu. 2001. Suspended sediment drift and dispersion on Georges Bank. In preparation for Can. Tech. Rep. Hydrog. Ocean Sci.

Lynch, D. R., J. T. C. Ip, C. E. Naimie, and F. E. Werner, 1996: Comprehensive coastal circulation model with application to the Gulf of Maine, *Contin. Shelf Res.*, **16**, 875-906.

Muschenheim, D. K., T. G. Milligan and D. C. Gordon. 1995. New technology and suggested methodologies for monitoring particulate wastes discharged from offshore oil and gas drilling platforms and their effects on benthic boundary layer environment. Can. Tech. Rep. Fish. Aquat. Sci. 2049: x + 55 p.

Muschenheim, D. K. and T. G. Milligan. 1996. Flocculation and accumulation of fine drilling waste particles on the Scotian Shelf. *Mar. Pollut. Bull.* **32**, 740–745.

Naimie, C. E. 1996. Georges Bank residual circulation during weak and strong stratification periods - Prognostic numerical model results, *J. Geophys. Res.* **101**, 6469-6486.

Preisendorfer, R. W. 1988. Principal Component Analysis in Meteorology and Oceanography. Elsevier. 425 pp.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. W. T. Vetterling. 1992. Numerical Recipes in Fortran 77. Second Edition. Cambridge University Press. 902 pp.

Xu, Z., C. G. Hannah and J. W. Loder. 2000. A 3D shear dispersion model applied to Georges Bank. In: Estuarine and Coastal Modeling: Proceedings of the 6th International Conference. 3-5 November, 1999 New Orleans. ASCE. Pages 581-596.

A Logarithmic Sublayer

The implementation of the logarithmic sublayer follows the formalism developed for the finite-element circulation model QUODDY which was used to generate the 3-d circulation fields for spatial `bblt` (Naimie, 1996; Lynch et al., 1996). Thus, the logarithmic sublayer used in `bblt` is consistent with the velocity profiles taken from the circulation model. It is also a reasonable representation for use with observed currents from the bottom few m. Understanding the representation of the logarithmic sublayer is important for `bblt` because most of the velocity shear in tidal flows is near the bottom, and it is the velocity shear which generates the dispersion in `bblt`.

Here we describe the details of the logarithmic sublayer implementation, using for illustrative purposes the simplified case of the instantaneous near-bottom flow $\mathbf{u_b}$ directed in the x-direction (or the x-axis rotated to the direction of $\mathbf{u_b}$).

QUODDY does not resolve the entire bottom boundary layer; rather, the bottom of the computational domain is assumed to be in the constant stress layer (log layer) and displaced a small distance above the ocean bottom. Closure is obtained via a quadratic stress law:

$$\frac{|\tau_b|}{\rho_0} = u_*^2 = C_d|\mathbf{u}_b|^2 \tag{10}$$

where τ_b is the bottom stress, ρ_0 is the reference density, u_* is the friction velocity, C_d is the drag coefficient and $\mathbf{u}_b$ is the velocity at the bottom of the model. (We deal with the inconsistency between (10) and (2) below.)

Let ξ be the distance above the seafloor, $u(\xi)$ the x-component of velocity, ξ_b the height of the bottom node above the seafloor and $u_b = u(\xi_b)$. The logarithmic profile is

$$u(\xi) = \frac{u_*}{\kappa}\ln(\xi/\xi_0) \tag{11}$$

where ξ_0 is the roughness height, and κ (= 0.4) the von Karman constant. The relationship between ξ_b, ξ_0 and C_d (found by substituting the velocity profile into the stress law) is

$$\xi_0 = \xi_b e^{-\kappa/\sqrt{C_d}} \tag{12}$$

In the present QUODDY solutions, $\xi_b = 1$ m and $C_d = 0.005$ which yields $\xi_0 = 0.003495$ m. Notice that ξ_0 is the height at which the velocity goes to zero, and could be interpreted to mark the transition to the viscous sublayer where horizontal velocities are assumed vanishingly small.

From the point of view of `bblt`, u_b is known so the velocity profile is computed using

$$u(\xi) = u_b\frac{\sqrt{C_d}}{\kappa}\ln(\xi/\xi_0) \quad \text{for } h_{ref} < \xi < \xi_b \tag{13}$$

which is obtained by substituting the stress law into the velocity profile. We also require $\xi_0 < h_{ref}$. With this relation, there is no rotation with height of the velocity in the logarithmic sublayer so that, in cases where ξ_b is more than a few m, the profile is only a crude approximation to the bottom Ekman layer.

The logarithmic sublayer presented is self-consistent. What happens if we adjust some of the relationships? In particular, in some cases we want to use a general stress law such as (2) to include additional stresses due to unmodelled velocity components or wave effects. For example, calculations of enhanced stress due to waves and wave-current interactions indicate that, for the water depths of interest on Georges Bank, bottom stress values can be 50–100% larger than those computed from a stress law like (10) (Anderson , 2001). However, the user may still need to take u_b and ξ_b from the 3-d model profile or moored measurements (to focus on velocity changes within the sublayer and the effects of the stress changes on sediment suspension). Under these conditions the velocity profile in the logarithmic sublayer can be written as

$$u(\xi) = u_b + \frac{u_*}{\kappa} \ln(\xi/\xi_b) \tag{14}$$

where $\xi \leq \xi_b$. The friction velocity u_* can now be freely specified (i.e. it is not constrained by (10)).

The consequences of such an adjustment are as follows. Specifying u_* greater than that implied by (10) has the effect of increasing the effective ξ_0, while decreasing u_* decreases ξ_0. In fact, factor of two changes in u_* imply order of magnitude changes in ξ_0. It is easy to generate cases where $\xi_0 > h_{ref}$ for reasonable values of h_{ref}. For $\xi_0 > h_{ref}$ the velocities $u(\xi)$ for $\xi < h_{ref}$ are the opposite sign of u_b. On the other hand, for $\xi_0 \ll h_{ref}$ most of the velocity shear occurs in the region $\xi < h_{ref}$ and there is very little shear dispersion near the bottom.

We have adopted the following procedure in order to avoid having changes in the stress law (2) (which is intended to control the vertical extent of the concentration profile) change the dynamical consistency of the logarithmic sublayer:

- The roughness height $\xi_0 = 0.003495$ m is fixed.
- The height above the bottom of the near-bottom current meter (or model node), ξ_b, is taken as specified by the user.
- The drag coefficient C_d is calculated from

$$C_d = (\kappa / \ln(\xi_b/\xi_0))^2 \tag{15}$$

  For $\xi_b = 1$ m $C_d = 0.005$, and for $\xi_b = 3$ m $C_d = 0.0035$.

- The velocities in the sublayer are calculated according to (13).

The velocity profile is discretized at user-specified levels ξ_s with $s = 1, N_{sl}$, and linear interpolation is used to estimate the velocity between these levels (in an identical manner to the specified velocities above the sublayer). If the user specifies $N_{sl} = 2$, then a linear sublayer is created.

Within `bblt` the calculation of the parameters for the logarithmic sublayer are independent of the calculation of u_* for the Rouse profile (2). This is consistent with our approach that the velocity profiles are specified and not changed by sediment processes. However, we have used the relation (15) to calculate the drag coefficient to use with the observed currents where the height above the bottom can range from 3 to 14 m.

The logarithmic sublayer has the feature that the velocity at the reference height is not zero, whereas it was zero in the linear sublayer in `bblt` v1. The velocity at the reference height is

$$u(h_{ref})/u_b = 1 - \ln h_{ref}/\ln \xi_0 \quad . \tag{16}$$

For the typical values of ξ_0 and h_{ref} ($\xi_0 = 0.003495$ m, $h_{ref} = 0.0035$ m) the velocity ratio $u(h_{ref})/u_b = 2.5 \times 10^{-4}$ and the velocity experienced by packets near the bottom is very small. Notice however that for $h_{ref} = 0.01$ m (a reasonable value and the one used in Hannah et al. 1995, 1996, 1998), $u(h_{ref})/u_b = 0.19$ which is quite large and indicates that most of the near bottom shear occurs for $\xi < h_{ref}$.

B List of Files

The complete list of source code files which constitute `bblt v3.1`

```
AbRVertPos.f
CalcUstar.f
CrtPckts2.f
InterpVel.f
MoveHoriz.f
PDFcompute.f
RdParams2.f
SedFcns.f
UserSetup.f
VelHoriz.f
VertShffl.f
bbltv31.f
nodeinfo2.f
output4b.f
ran1.f   --- not part of the distribution.
velinfo3.f
velinfo5.f

UserSetup.h
bblran.h
bbltv20.h
```

C Main Module

The source code for the main module of `bblt v3.1`.

```
C*************************************************
C  Copyright  Government of Canada
C             Department of Fisheries and Oceans
C             Bedford Institute of Oceanography
C             Dartmouth, N.S.
C*************************************************
C
C*** bbltmainv30.f
C
C* benthic boundary layer transport model Version 3
C  Continuous release.
C
C* main module
C
C  Initialization
C     User Setup
C     Read Input Parameters (including random number seed)
C     Compute timing parameters
C     Get Initial Packet Positions
C     Initialize Vertical Shuffling
C     Get Vertical Node Information
C     Get Initial Velocity
C     Prepare for Output
C     Stop if errors were reported
C
C  Time Stepping (Loop over velocity updates)
C     Get Velocity information at the end of velocity update interval
C     Loop over advection time steps
C        Compute average velocity over advection time step
C        Compute and other parameters bottom stress
C        Create more Packets (if desired)
C        Vertical Shuffling
C        Compute Vertical Probability Distribution
C        Compute Vertical Positions
C        Horizontal Advection
C        Call Output
C
C  Shutting down
C     Call Output with all_done flag set.
C
C
```

```
C*********************************************************************
       program bbltr

      implicit none

      include 'bbltv20.h'
      include 'bblran.h'

c***Packets
      real    x(kmax), y(kmax), vertpos(kmax)
      real    ranks(kmax)
      integer ktotal
      real    mtotal
      real    pmass
      common  /packets/x,y,vertpos,ranks,ktotal,pmass

c***Velocity
      real    u(maxnodes), v(maxnodes)
      real    u1(maxnodes), v1(maxnodes)
      real    u2(maxnodes), v2(maxnodes)
      real    heightnodes(maxnodes)
      integer nloglayer, ninterior
      integer nnv
      common  /velocity/u,v,heightnodes,nnv,nloglayer,ninterior


c***Vertical Structure and Sediment Parameters
      real    ustar, crustar
      real    suspfrac
      real    pRouse
      real    PDFvi(nmaxPDF,2)
      integer nentry

      real    kappa
      integer nfricp,nfricmax
      parameter (nfricmax = 3)
      real    friction(nfricmax)
      real    wstar
      common  /sediment/wstar,ustar, crustar,pRouse,suspfrac,
     $                  PDFvi,nentry

c***Time
      real    dta,dtap,dtvu, dtms
      real    apu
      real    ttotal,time0
```

```
      real    tnextupdate
      real    time1
      integer velupdates
      integer istepn
      real    timec,timef
      real    ctime
      real    fracmix

      common  /timing/dtap,dtvu, dtms,apu,ttotal,time0,timec,timef

c***Misc
c     real    rho
      real    href
      real    hmax
      integer vmethod

c***Flags
      integer nerror
      integer conditionflag, errorcode
C
C condition flag < 0 implied initialization
C                = 0 normal
C                > 0 run completed

c***Functions
      real    CalcUstar
      real    Threshold
      real    SuspendedFraction
      real    Rousenum


      integer  i,j
      real     z
      character*120 echofile

      errorcode = 0
      conditionflag = -1
      nerror = 0

C
C**********************Initialization*****************************
C
      write(*,*)'Enter file name for error messages and ',
     $          'misc. info messages'
      read(*,'(a)')echofile
```

```
      open(2,file=echofile,status='unknown')
      write(2,*)'***Echo file for bblt v2.0***'
      write(2,*)' '

      write(*,*)'Enter random number seed (suggest 8-9 digits)'
      read(*,*)idum
      write(2,*)'Random number seed'
      write(2,*)idum
      if(idum.gt.0) idum = -idum

C***User Setup
      call UserSetup(errorcode)
      if(errorcode.ne.0) then
          nerror = nerror + 1
          call ErrorReport(errorcode,'UserSetup')
      endif

C***Read Input Parameters (including random number seed)
      call RdParams(dta,dtms,dtvu,ttotal,time0,
     $      wstar,kappa,href,friction,nfricp,nfricmax,hmax,
     $      vmethod,errorcode,debug)
      if(errorcode.ne.0) then
          nerror = nerror + 1
          call ErrorReport(errorcode,'ReadInputParameters')
      endif

      if(debug) then
         write(2,*)
         write(2,*)'ReadInputParameters:'
         write(2,*)'  dta,dtvu,ttotal,time0 ', dta,dtvu,ttotal,time0
         write(2,*)'  dtms ', dtms
         write(2,*)'  wstar,kappa,href,hmax ',
     $               wstar,kappa,href,hmax
         write(2,*)'  friction ',(friction(i), i = 1,nfricp)
         write(2,*)'  vmethod,errorcode ',vmethod,errorcode
      endif

C***Compute timing parameters and determine the fraction of packets that
C       get shuffled every time step.
      velupdates = int(ttotal/dtvu)
      z=dtvu/dta
      apu=nint(z)
      dtap=dtvu/float(apu)
      fracmix=dtap/dtms
      write(2,*)
```

```
      write(2,*)'Timing information.'
      write(2,*)' # of velocity updates ', velupdates
      write(2,*) 'z,dtms ', z, dtms
      write(2,*) 'actual advection time step ',
     $              '(time between shuffling events) ', dtap
      write(2,*) 'advection steps per velocity update', apu
      write(2,*) 'fraction shuffled per advection step', fracmix
      write(2,*) 'fraction shuffled per velocity update',
     $           fracmix*apu

C***Get Initial Packet Positions
      call CrtPckts2(vmethod,x,y,vertpos,ranks,kmax,
     $          ktotal,time0,dtap,mtotal,pmass,
     $          debug,conditionflag,errorcode)

      if(errorcode.ne.0) then
          nerror = nerror + 1
          call ErrorReport(errorcode,'CreatePackets')
      endif

C***Initialize Vertical Shuffling
      call VertShuff1(vmethod,ranks,ktotal,kmax,fracmix,
     $                     conditionflag,errorcode,debug)
      if(errorcode.ne.0) then
          nerror = nerror + 1
          call ErrorReport(errorcode,'VerticalShuffle')
      endif

C***Get Vertical Node Information
      call NodeInfo(nloglayer,ninterior,nnv,
     $                    heightnodes,hmax,href,
     $                    errorcode)
      if(errorcode.ne.0) then
          nerror = nerror + 1
          call ErrorReport(errorcode,'GetVertNodeInfo')
      endif
      if (nloglayer+ninterior.gt.maxnodes) then
         write(*,*)
         write(*,*)'GetVertNodeInfo requires too many vertical nodes.'
         write(*,*)'nloglayer,ninterior,nnv ', nloglayer,ninterior,nnv
         write(*,*)'Maximum # specified in parameter statement ',maxnodes
         write(2,*)
         write(2,*)'GetVertNodeInfo requires too many vertical nodes.'
         write(2,*)'  nloglayer,ninterior,nnv ', nloglayer,ninterior,nnv
         write(2,*)'  Maximum # specified in parameter statement ',
```

```
     $              maxnodes
            nerror = nerror + 1
          endif

C
C***Get Initial Velocity
          istepn = 0
          call VelInfo(nnv,u2,v2,time0,dtvu,istepn,
     $              nloglayer,ninterior,heightnodes,
     $              errorcode)

          if(errorcode.ne.0) then
              nerror = nerror + 1
              call ErrorReport(errorcode,'GetVelInfo')
          endif
C                        so initial velocity available to output
          do j = 1,nnv
             u(j) = u2(j)
             v(j) = v2(j)
          enddo

C***Prepare for Output
          call Output(conditionflag,errorcode)

          if(errorcode.ne.0) then
              nerror = nerror + 1
              call ErrorReport(errorcode,'Output')
          endif

C***Check the number of errors reported
          if(nerror.ne.0) then
             write(2,*)
             write(2,*)' Number of errors reported in initialiation ',
     $                  nerror
             write(2,*)'Too many to carry on.  Time to die'
             write(*,*)
             write(*,*)' Number of errors reported in initialiation ',
     $                  nerror
             write(*,*)' Further details are given in file '
             write(*,'(a)')echofile
             write(*,*)'Program will now end its life'
             STOP
          endif

          write(2,*)'Hmax = ',hmax
```

```
C
C*****************************Time Stepping**********************
C
       conditionflag = 0

C***Loop over velocity updates
       do i = 1,velupdates
          time1 = (i-1)*dtvu + time0
          tnextupdate = time1 + dtvu
          do j = 1,nnv
              u1(j) = u2(j)
              v1(j) = v2(j)
          enddo

C***Get Velocity information at the end of velocity update interval
          call VelInfo(nnv,u2,v2,tnextupdate,dtvu,i,
     $                    nloglayer,ninterior,heightnodes,
     $                    errorcode)
          if(errorcode.ne.0) then
             nerror = nerror + 1
             call ErrorReport(errorcode,' GetVelInfo')
          endif

C*** Loop over advection time steps
          do j=1,apu
             ctime = (j-0.5)*dtap
             timec  = ctime + time1
             timef  = timec + 0.5*dtap
C***Compute average velocity over advection time step
             call InterpVel(u,u1,u2,nnv,ctime,dtvu)
             call InterpVel(v,v1,v2,nnv,ctime,dtvu)

C***Compute and other parameters bottom stress
             ustar = CalcUstar(u(nloglayer),v(nloglayer),
     $                         friction,nfricp,debug,errorcode)
             if(errorcode.ne.0) then
                 call ErrorReport(errorcode,' CalcUstar')
                 nerror = nerror + 1
             endif

             crustar = Threshold()
             suspfrac = SuspendedFraction(ustar,crustar)
             pRouse = Rousenum(wstar,kappa,ustar)
C
```

```
C***Create more Packets (if desired)
            call CrtPckts2(vmethod,x,y,vertpos,ranks,kmax,
     $                     ktotal,timec,dtap,mtotal,pmass,
     $                     debug,conditionflag,errorcode)
            if(errorcode.ne.0) then
               call ErrorReport(errorcode,' CreatePackets')
               nerror = nerror + 1
            elseif(ktotal.gt.kmax) then
               write(*,*)'**Fatal Error**.  Too many packets'
               write(2,*)'**Fatal Error**.  Too many packets'
               write(2,*)'ktotal, kmax ',ktotal, kmax
               stop
            endif

C***Vertical Shuffling
C***Compute Vertical Probability Distribution
C***Compute Vertical Positions
             if(vmethod.ne.1) then
                call VertShuffl(vmethod,ranks,ktotal,kmax,
     $             fracmix,conditionflag,errorcode,debug)
                if(errorcode.ne.0) then
                   call ErrorReport(errorcode,' VerticalShuffle')
                   nerror = nerror + 1
                endif
                call PDFcompute(PDFvi,nentry,nmaxPDF,suspfrac,pRouse,
     $                          href,hmax)
                call AbRVertPos(vertpos,ranks,ktotal,PDFvi,
     $                          nmaxPDF,nentry,href,hmax)
             endif

C***Horizontal Advection
             call MoveHoriz(x,y,vertpos,ktotal,dtap,nnv,
     $                       heightnodes, u,v)

C***Output
             call Output(conditionflag,errorcode)
             if (errorcode.ne.0) then
                call ErrorReport(errorcode,' VerticalShuffle')
                nerror = nerror + 1
             endif
             if(nerror.ne.0) goto 1001
          enddo
       enddo
1001   continue
C
```

```
C*************************Clean Up and Exit*******************
C
      conditionflag = 1
      call Output(conditionflag,errorcode)

      if(nerror.eq.0) then
         write(*,*)' Clean exit '
         write(2,*)' Clean exit '
      else
         write(*,*)'***Exit on error***'
         write(*,*)'See the log file for more information.'
         write(2,*)'***Exit on error***'
         write(2,*)'i,timef, nerror, errorcode = ',
     $              i,timef, nerror,errorcode
      endif
      stop
      end
```